AMENDMENTS TO THE CLAIMS

The listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for compiling source code for a plurality of heterogeneous processor types, said method comprising:

receiving source code that includes a plurality of source code subtasks;

independently selecting a processor type from the plurality of heterogeneous processor types for each of the plurality of source code subtasks, the independent selection comprising:

selecting a first processor type from the plurality of heterogeneous processor types for a first source code subtask included in the source code; and

selecting a second processor type from the plurality of heterogeneous processor types for a second source code subtask included in the source code, wherein the second processor type is different than the first processor type; and

after the processor type selections, compiling the source code, which directly creates ~~creating a~~ an object file that includes a first object code corresponding to the first source code subtask and a second object code corresponding to the second source code subtask, wherein the first object code is adapted to be processed by the first processor type and the second object code is adapted to be processed by the second processor type.

2. (Canceled)

3. (Previously Presented) The method as described in claim 1 wherein the selection of the first processor type is performed during compilation, the method further comprising:

retrieving the first source code subtask from the plurality of source code subtasks;

determining whether the first source code subtask includes a program directive corresponding to one of the plurality of processors; and

performing the selection of the first processor type in response to the determination.

4. (Previously Presented) The method as described in claim 1 further comprising:

retrieving the first source code subtask from the plurality of source code subtasks; and

compiling the first source code subtask, the compiling resulting in byte code.

5. (Original) The method as described in claim 4 further comprising:

sending the byte code to a client over a computer network, wherein the byte code is adapted to be translated into client-specific object code by the client whereby the client-specific object code is formatted based upon a processor type that is located at the client.

6. (Previously Presented) The method as described in claim 1 further comprising:

retrieving the first source code subtask from the plurality of source code subtasks;

identifying one or more operations included in the first source code subtask;

matching one or more of the operations with one of the processor types from the plurality of heterogeneous processor types; and

performing the selection of the first processor type in response to the matching.

7. (Previously Presented) The method as described in claim 1 further comprising:

receiving a processor-specific command, the processor specific command identifying a processor type from the plurality of heterogeneous processor types; and

performing the selection of the first processor type based upon the processor-specific command.

8. (Currently Amended) An information handling system comprising:

a plurality of heterogeneous processors;

a memory accessible by the heterogeneous processors;

one or more nonvolatile storage devices accessible by the heterogeneous processors; and

a source code compilation tool for compiling source code, the source code compilation tool comprising software code effective to:

receive source code that includes a plurality of source code subtasks from one of the nonvolatile storage devices;

independently select a processor type from the plurality of heterogeneous processor types for each of the plurality of source code subtasks comprising:

select a first processor type from the plurality of heterogeneous processor types for a first source code subtask included in the source code; and

select a second processor type from the plurality of heterogeneous processor types for a second source code subtask included in the

source code, wherein the second processor type is different than the first processor type; and

after the processor type selections, compile the source code, which directly creates ~~create~~ an object file that includes a first object code corresponding to the first source code subtask and a second object code corresponding to the second source code subtask, wherein the first object code is adapted to be processed by the first processor type and the second object code is adapted to be processed by the second processor type.

9. (Canceled)

10. (Previously Presented) The information handling system as described in claim 8 wherein the selection of the first processor type is performed during compilation, wherein the software code is further effective to:

retrieve the first source code subtask from the plurality of source code subtasks located in one of the nonvolatile storage devices;

determine whether the first source code subtask includes a program directive corresponding to one of the plurality of processors; and

perform the selection of the first processor type in response to the determination.

11. (Previously Presented) The information handling system as described in claim 8 wherein the software code is further effective to:

retrieve the first source code subtask from the plurality of source code subtasks; and

compile the first source code subtask, the compiling resulting in byte code.

12. (Original) The information handling system as described in claim 11 wherein the software code is further effective to:

send the byte code to a client over a computer network, wherein the byte code is adapted to be translated into client-specific object code by the client whereby the client-specific object code is formatted based upon a processor type that is located at the client.

13. (Previously Presented) The information handling system as described in claim 8 wherein the software code is further effective to:

retrieve the first source code subtask from the plurality of source code subtasks located in one of the nonvolatile storage devices;

identify one or more operations included in the first source code subtask;

match one or more of the operations with one of the processor types from the plurality of heterogeneous processor types; and

perform the selection of the first processor type in response to the matching.

14. (Currently Amended) A computer program product stored on a computer operable media, the computer operable media containing instructions for execution by a computer, which, when executed by the computer, cause the computer to implement a method for compiling source code for a plurality of heterogeneous processors, the method comprising:

receiving source code that includes a plurality of source code subtasks;

independently selecting a processor type from the plurality of heterogeneous processor types for each of the plurality of source code subtasks, the independent selection comprising:

selecting a first processor type from the plurality of heterogeneous processor types for a first source code subtask included in the source code; and

selecting a second processor type from the plurality of heterogeneous processor types for a second source code subtask included in the source code, wherein the second processor type is different than the first processor type; and

after the processor type selections, compiling the source code, which directly creates ~~creating~~ an object file that includes a first object code corresponding to the first source code subtask and a second object code corresponding to the second source code subtask, wherein the first object code is adapted to be processed by the first processor type and the second object code is adapted to be processed by the second processor type.

15. (Canceled)

16. (Previously Presented) The computer program product as described in claim 14 wherein the selection of the first processor type is performed during compilation, the method further comprising

retrieving the first source code subtask from the plurality of source code subtasks;

determining whether the first source code subtask includes a program directive corresponding to one of the plurality of processors; and

performing the selection of the first processor type in response to the determination.

17. (Previously Presented) The computer program product as described in claim 14 wherein the method further comprises :

retrieving the first source code subtask from the plurality of source code subtasks; and

compiling the first source code subtask, the compiling resulting in byte code.

18. (Previously Presented) The computer program product as described in claim 17 wherein the method further comprises :

sending the byte code to a client over a computer network, wherein the byte code is adapted to be translated into client-specific object code by the client whereby the client-specific object code is formatted based upon a processor type that is located at the client.

19. (Previously Presented) The computer program product as described in claim 14 wherein the method further comprises :

retrieving the first source code subtask from the plurality of source code subtasks;

identifying one or more operations included in the first source code subtask;

matching one or more of the operations with one of the processor types from the plurality of heterogeneous processor types; and

performing the selection of the first processor type in response to the matching.

20. (Previously Presented) The computer program product as described in claim 14 wherein the method further comprises :

receiving a processor-specific command, the processor specific command identifying a processor type from the plurality of heterogeneous processor types; and

performing the selection of the first processor type based upon the processor-specific command.